UDACITY

PROJECT SPECIFICATION

Capstone, Stage 2 - Build

Common Project Requirements

MEETS SPECIFICATIONS

App conforms to common standards found in the Android Nanodegree General Project Guidelines

App is written solely in the Java Programming Language

App utilizes stable release versions of all libraries, Gradle, and Android Studio.

Core Platform Development

MEETS SPECIFICATIONS

App integrates a third-party library.

MEETS SPECIFICATIONS

App validates all input from servers and users. If data does not exist or is in the wrong format, the app logs this fact and does not crash.

App includes support for accessibility. That includes content descriptions, navigation using a D-pad, and, if applicable, non-audio versions of audio cues.

App keeps all strings in a `strings.xml` file and enables RTL layout switching on all layouts.

App provides a widget to provide relevant information to the user on the home screen.

Google Play Services

MEETS SPECIFICATIONS

App integrates two or more Google services. Google service integrations can be a part of Google Play Services or Firebase.

Each service imported in the `build.gradle` is used in the app.

If `Location` is used, the app customizes the user's experience by using the device's location.

MEETS SPECIFICATIONS

If `Admob` is used, the app displays test ads. If `Admob` was not used, student meets specifications.

If `Analytics` is used, the app creates only one analytics instance. If `Analytics` was not used, student meets specifications.

If `Maps` is used, the map provides relevant information to the user. If `Maps` was not used, student meets specifications.

If `Identity` is used, the user's identity influences some portion of the app. If `Identity` was not used, student meets specifications.

Material Design

MEETS SPECIFICATIONS

App theme extends `AppCompat`.

App uses an app bar and associated toolbars.

App uses standard and simple transitions between activities.

Building

MEETS SPECIFICATIONS

App builds from a clean repository checkout with no additional configuration.

App builds and deploys using the `installRelease` `Gradle` task.

App is equipped with a signing configuration, and the keystore and passwords are included in the repository. Keystore is referred to by a relative path.

All app dependencies are managed by `Gradle`.

Data Persistence

MEETS SPECIFICATIONS

App stores data locally either by implementing a ContentProvider OR using Firebase Realtime Database OR using Room. No third party frameworks nor Persistence Libraries may be used.

MEETS SPECIFICATIONS

Must implement at least **one** of the three:
If it regularly pulls or sends data to/from a web service or API, app updates data in its cache at regular intervals using a SyncAdapter or JobDispatcher.
OR
If it needs to pull or send data to/from a web service or API only once, or on a per request basis (such as a search application), app uses an IntentService to do so.
OR
It it performs short duration, on-demand requests(such as search), app uses an AsyncTask.

If Content provider is used, the app uses a Loader to move its data to its views.

If Room is used then LiveData and ViewModel are used when required and no unnecessary calls to the database are made.

Suggestions to Make Your Project Stand Out!

- Make your app more delightful with material design patterns such as **shared element transitions** across activities and **parallax scrolling** where two or more items must scroll in the same activity.
- Implement **notifications** in your app. Remember the following when implementing notifications:
    - Notifications should not contain advertising or content unrelated to the core function of the app.
    - Notifications should be persistent only if related to ongoing events (such as music playback or a phone call).
    - Multiple notifications are stacked into a single notification object, where possible.
    - Use notifications only to indicate a context change relating to the user personally (such as an incoming message).
    - Use notifications only to expose information/controls relating to an ongoing

event (such as music playback or a phone call).

- Implement **sharing** functionality in your app, making use of intent extras to share rich content (i.e. a paragraph of content-specific text, a link and description, an image, etc).
- Create and use a **custom view** in your app that could not be achieved with the standard widgets provided by the core views on Android.